Relatório Técnico: Sistema de Publicadores e Assinantes com Broker

# 1. Objetivo do Código

Este código implementa um sistema simples de publicação e assinatura (publish/subscribe), onde publicadores enviam mensagens para tópicos específicos e assinantes recebem essas mensagens através de um intermediário (broker). O objetivo principal é simular a comunicação assíncrona e distribuída entre várias entidades, utilizando threads e filas em Python para gerenciar a troca de mensagens entre publicadores e assinantes.

# 2. Componentes Principais

O sistema é composto por três funções principais:

## Publicadores (Publishers)

Publicadores são responsáveis por criar e enviar mensagens para um tópico selecionado aleatoriamente de uma lista fixa de tópicos: esportes, noticias, tecnologia. Cada publicador envia um número limitado de mensagens (15 mensagens neste caso) e utiliza uma fila (Queue) compartilhada para enviar as mensagens para o broker.

## Assinantes (Subscribers)

Assinantes escolhem um tópico de forma aleatória no qual desejam se inscrever. Eles ficam em espera para receber mensagens relacionadas ao tópico de assinatura. A comunicação entre assinantes e o broker é feita por meio de uma fila separada para cada assinante. Se o assinante não receber mensagens por 60 segundos, sua assinatura é cancelada.

## Broker

O broker atua como o intermediário entre publicadores e assinantes. Ele monitora a fila de mensagens enviadas pelos publicadores e, sempre que uma mensagem é postada, distribui-a aos assinantes que estão inscritos no tópico correspondente. Se não houver assinantes para um determinado tópico, a mensagem é descartada. O broker encerra sua execução quando todas as threads de publicadores e assinantes terminam.

# 3. Descrição do Fluxo de Execução

## Publicação de Mensagens

Cada publicador inicia sua execução em uma thread separada. Eles entram em um loop, onde, a cada 2 segundos, escolhem aleatoriamente um tópico, criam uma mensagem e enviam essa mensagem para a fila de mensagens (Queue) compartilhada.

## Assinatura de Tópicos

Os assinantes escolhem um tópico de forma aleatória e se inscrevem nele. Para cada assinatura, é criada uma fila individual onde o broker poderá enviar mensagens. Enquanto o assinante está ativo, ele verifica a fila associada para ver se há novas mensagens. Se não houver mensagens por 60 segundos, o assinante considera que houve um 'timeout' e cancela sua assinatura.

## Distribuição de Mensagens (Broker)

O broker verifica constantemente a fila de mensagens compartilhada pelos publicadores. Quando uma nova mensagem é postada, ele verifica quais assinantes estão inscritos no tópico daquela mensagem e encaminha a mensagem para cada um dos assinantes relevantes. Se a fila de mensagens estiver vazia, o broker espera por 2 segundos antes de tentar novamente.

## Encerramento

O broker monitora o número de threads ativas no sistema. Quando restam apenas a thread principal e a do broker, o broker encerra sua execução, sinalizando o fim do programa.

# 4. Estrutura de Dados Utilizada

## Queue (Fila)

Utilizada para armazenar as mensagens que os publicadores enviam e também para gerenciar o canal de comunicação entre o broker e os assinantes. A Queue é uma estrutura thread-safe, o que significa que várias threads podem acessar e modificar as filas simultaneamente sem causar condições de corrida.

## Dicionários para Assinantes

list\_of\_subscribers: Dicionário que mapeia cada tópico para uma lista de assinantes inscritos naquele tópico.

subscribers\_channels: Dicionário que mapeia cada assinante para sua respectiva fila de comunicação, usada para receber as mensagens do broker.

# 5. Vantagens e Limitações

## Vantagens

- Concorrência com Threads: O uso de threads permite que múltiplos publicadores e assinantes operem simultaneamente, sem bloqueios, o que simula um ambiente de comunicação assíncrona.

- Uso de Fila para Comunicação Segura: A Queue é uma estrutura que facilita a comunicação entre threads de forma segura, garantindo que não haja corrupção de dados, mesmo com múltiplos produtores (publicadores) e consumidores (broker e assinantes).

## Limitações

- Atribuição de Tópicos de Forma Aleatória: Os tópicos são escolhidos de forma aleatória, tanto para os publicadores quanto para os assinantes. Isso pode gerar um cenário em que mensagens são enviadas para tópicos sem assinantes, resultando em desperdício de mensagens.

- Controle de Assinantes: Uma vez que um assinante cancela sua assinatura após o timeout, ele não pode se reinscrever. Seria interessante ter a opção de permitir que os assinantes renovassem suas assinaturas ou se inscrevessem em múltiplos tópicos.

- Encerramento Baseado no Número de Threads: O broker encerra sua execução quando apenas ele e a thread principal estão ativas. Um método mais explícito de controle, como um sinal de término de execução, seria mais robusto.

# 6. Possíveis Melhorias

1. Gerenciamento de Timeout dos Assinantes: Permitir que os assinantes se reinscrevam automaticamente após o timeout ou ajustarem o tempo limite dinamicamente, com base na frequência de mensagens recebidas.

2. Assinaturas Múltiplas: Atualmente, cada assinante pode se inscrever em apenas um tópico. Seria interessante implementar a possibilidade de assinantes se inscreverem em múltiplos tópicos simultaneamente.

3. Melhoria no Controle de Encerramento: O broker poderia utilizar um sinal explícito de encerramento, em vez de contar threads ativas. Isso permitiria mais controle sobre o ciclo de vida do sistema.

4. Persistência de Mensagens: Mensagens poderiam ser armazenadas temporariamente, permitindo que assinantes recebam mensagens antigas ao se inscreverem, caso tenham perdido alguma postagem.

# 7. Conclusão

Este código é um exemplo eficiente e simples de um sistema de comunicação publish/subscribe utilizando threads e filas em Python. Ele permite a comunicação assíncrona entre várias entidades (publicadores e assinantes) e é um ponto de partida sólido para sistemas de mensageria mais complexos. Com algumas melhorias, ele pode ser expandido para suportar mais funcionalidades e oferecer maior controle sobre o fluxo de mensagens e o gerenciamento de assinantes.